ARDUINO TEMPERATURE SYSTEM SIMULATION ON FAN

ASSIGNMENT 1

BY YI JIE LIM 20104720

9/4/2024, DOCUMENT REVISION V1.0

Bachelor of Science (Honours) in Computer Science
Automotive and Automation Systems Stream

TABLE OF CONTENTS

a) Title Page.. PG 1

b) Introduction.. PG 3

c) High-level Description/Design... PG 4

d) Low-level Description/Design...

i) Statechart + Description... PG 5

e) Implementation Description..

i) Code + Comments..PG 6 & 7 & 8 & 9 & 10

ii) Issues when doing the assignment... PG 11

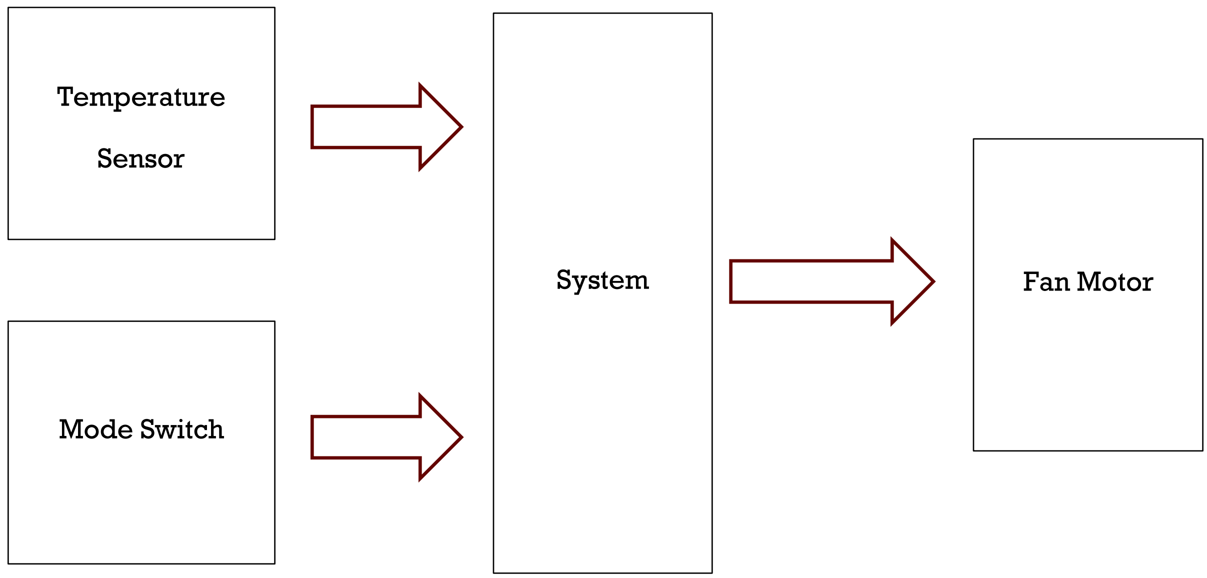
INTRODUCTION

This temperature system serves as an advanced temperature-based control mechanism fan. It's made up of an Arduino Uno R3, a TMP36 temperature sensor, a slideswitch, a voltage multimeter, a DC motor, a TIP120 transistor, and a 1 kΩ resistor.

The purpose for creating this system is to enhance the comfort and energy efficiency of the user's living space. By intelligently monitoring and regulating room temperature, the application ensures optimal conditions, providing a more comfortable environment to the user while contributing to energy savings.

The temperature control system's key characteristics and non-functional needs are listed in the customer's feature list. In order to keep consumers informed, the system must first offer real-time temperature monitoring, showing the current room temperature in degrees Celsius. Furthermore, it needs to have automatic temperature management, which would dynamically modify the fan motor speed in response to the detected temperature, guaranteeing the highest possible comfort levels. Prioritizing power efficiency to allow for possible battery-powered operation and putting safety safeguards in place to ensure appropriate system functioning, especially when regulating heating or cooling devices, are examples of non-functional needs. Additionally, the user interface must be fast, responding quickly to adjustments made by the user and temperature variations. Strongness and dependability are essential for maintaining stable functioning, reliable temperature regulation, and no malfunctions. Finally, an energy-efficient design is required to reduce ecological footprint and support sustainable practices due to environmental impact concerns.

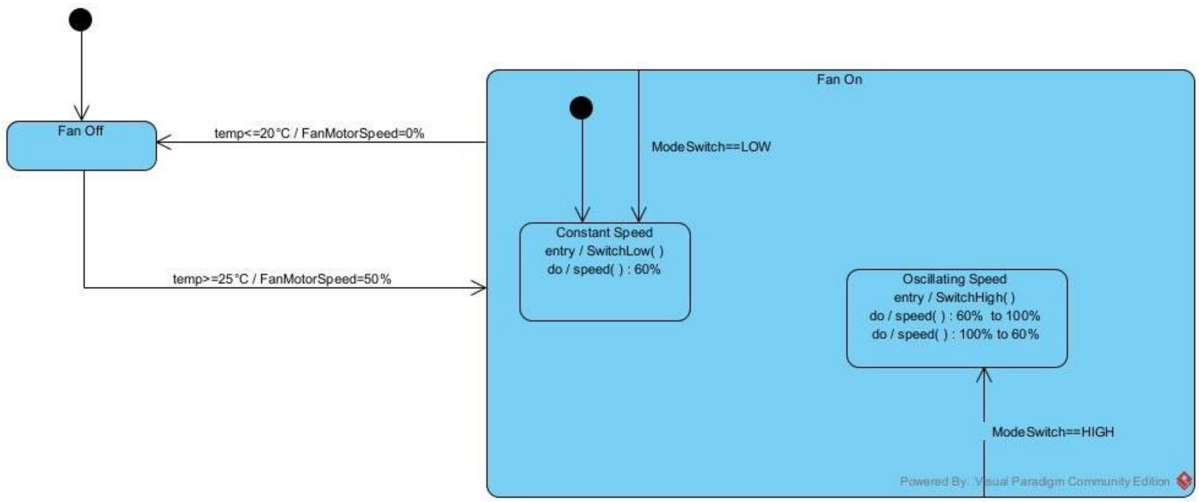
HIGH-LEVEL DESCRIPTION/DESIGN



The mode switch is functioning as a digital input device. It allows users to toggle between two distinct modes: constant speed or oscillating speed for the fan. Besides that, the temperature sensor serves as an input device as well. It's capturing analog data to measure the current temperature of the room. Additionally, the fan motor is connected as an output component and controlled by pulse width modulation (PWM). It allows fan speed variations based on the mode selected.

LOW-LEVEL DESCRIPTION/DESIGN

STATECHART + DESCRIPTION



The system operates based on two main events: temperature sensing and mode switching. When the temperature sensor detects a temperature of 25 degrees Celsius or higher, the system automatically activates the fan, ensuring optimal cooling conditions. Conversely, if the temperature falls below 20 degrees Celsius, the fan remains inactive unless manually turned on by the user. Additionally, the mode switch provides control over the fan speed, offering users the choice between a continuously on constant speed or an oscillating speed, depending on their preference for airflow direction and intensity. This integrated approach allows for efficient temperature regulation and customizable fan operation, enhancing comfort and energy efficiency in the living space.

IMPLEMENTATION DESCRIPTION

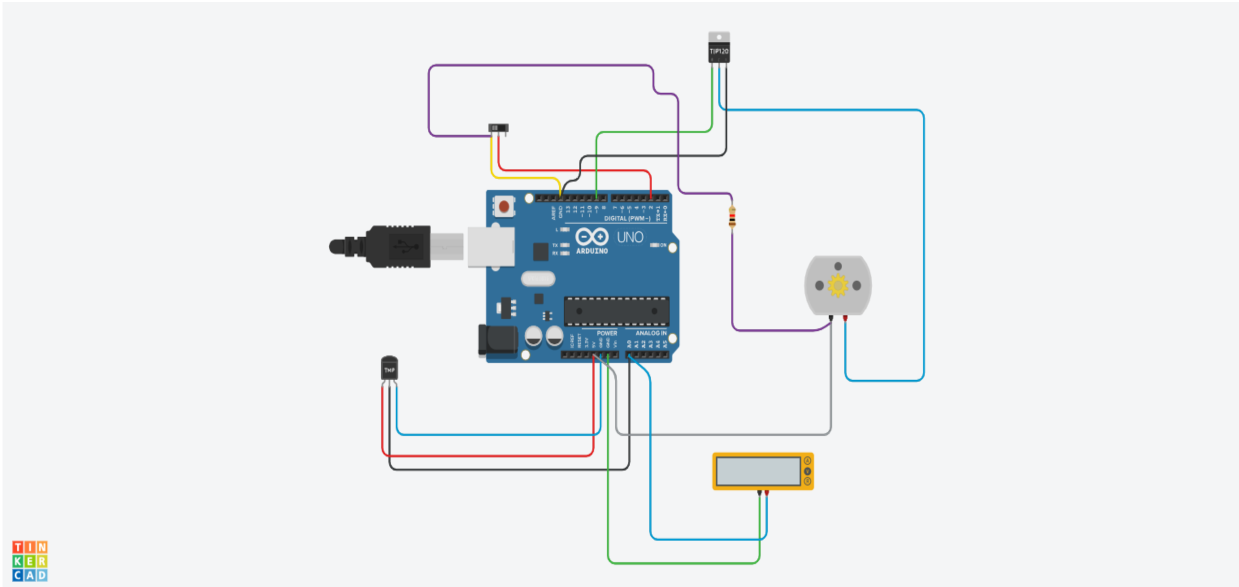
CODE + COMMENTS

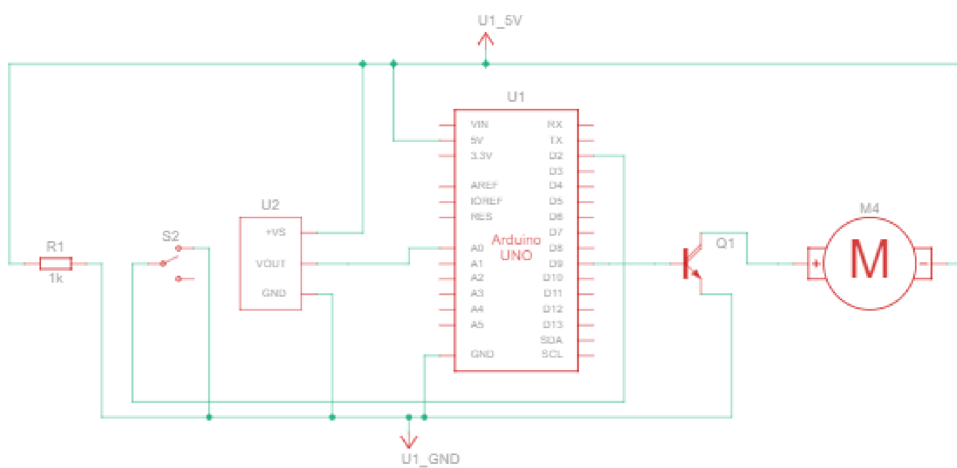
```
// Define pin numbers for mode switch, temperature sensor, and motor control

const int modeSwitchPin = 2;


const int temperaturePin = A0;


const int motorControlPin = 9;


// Define variables for motor speed, temperature thresholds, mode switch state, and current temperature


int motorSpeed = 0;


int motorConstantSpeed = 60;


int motorMaxSpeed = 100;


int tempFanOn = 157;

int tempFanOff = 153;

int modeSwitchState = 0;

float temperature = 0;
```

```
void setup () {
  // Begin serial communication
  Serial.begin(9600);

  // Set mode switch pin as input with internal pull-up
resistor
  pinMode (modeSwitchPin, INPUT_PULLUP);

  // Set motor control pin as output
  pinMode (motorControlPin, OUTPUT);
}

// Main loop
void loop () {
  // Read the state of the mode switch
  modeSwitchState = digitalRead(modeSwitchPin);

  // Read temperature sensor value
  int sensorValue = analogRead(temperaturePin);

  // Convert sensor value to temperature
  temperature = sensorValue;
```

```
  // Check if temperature is above the threshold for fan
activation

  if (temperature >= tempFanOn) {

    // If mode switch is LOW (constant speed mode)

    if (modeSwitchState == LOW) {

      // Set motor speed to constant speed

      analogWrite (motorControlPin,motorConstantSpeed);

      delay (10);

    }

    // If mode switch is HIGH (oscillating speed mode)

    if (modeSwitchState == HIGH) {

      // Increase motor speed gradually from constant
speed to maximum speed

      for (int speed = motorConstantSpeed; speed <=
motorMaxSpeed; speed++) {

        analogWrite (motorControlPin, speed);

        delay (10);

      }

      // Decrease motor speed gradually from maximum
speed to constant speed

      for (int speed = motorMaxSpeed; speed >=
motorConstantSpeed; speed--) {

        analogWrite(motorControlPin, speed);

        delay (10);

      }
```

```
    }
  }
  // Check if temperature is below the threshold for
fan deactivation
  if (temperature <= tempFanOff) {
    // Turn off the motor
    analogWrite (motorControlPin, 0);
    delay (10);
  }
}
```




U1_5V
U1
VIN
5V
3.3V
AREF
IOREF
RES
A0
A1
A2
A3
A4
A5
GND
RX
TX
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
D13
SDA
SCL
Arduino
UNO
U2
+VS
VOUT
GND
R1
1k
S2
Q1
M4
M
U1_GND

ISSUES WHEN DOING THE ASSIGNMENT

I ran across several problems with these three tasks during the assignment procedure. To begin with, I have no idea which wire has to be linked to which plot in order for this circuit to function. Luckily, Brendan, my instructor, helped me finish it. Second, I discovered some mistakes in my code that prevented my DC motor from operating. Brendan was able to assist me, and eventually my circuit functioned, marking the completion of my task. I gained a lot of knowledge from this homework that I was not previously aware of. For example, designing a circuit and writing code to create an embedded system. I'd want to thank Brendan and myself for helping me complete this homework.